

Paulo Marcos Dores Junior

# **SISLETRON: uma Plataforma Web de Auxílio ao Letramento Infantil**

**Curitiba**

**2015**



Paulo Marcos Dores Junior

# **SISLETRON: uma Plataforma Web de Auxílio ao Letramento Infantil**

Trabalho de Graduação apresentado como  
requisito parcial à obtenção do grau de Ba-  
charel em Ciência da Computação pelo Setor  
de Ciência Exatas da Universidade Federal  
do Paraná

Universidade Federal do Paraná – UFPR

Departamento de Informática

Orientador: Bruno Muller Junior

Curitiba

2015

# Resumo

Este trabalho teve como objetivo implementar uma plataforma web para letramento infantil utilizando o framework de desenvolvimento web Ruby on Rails. Neste sistema, o professor usuário do sistema é capaz de propor projetos, criar e modificar exercícios e materiais motivadores relacionados ao contexto da aprendizagem dos alunos. E estes, por sua vez, são capazes de ler os materiais e responder os exercícios relacionados ao mesmo.

**Palavras-chaves:** plataforma web, letramento.

# Abstract

This work aimed to develop a learning web platform for children using the Ruby on Rails web development framework. The professor through the system is capable of proposing projects, create and modify exercises and motivational materials related to the context of the student learning environment. The students are able to read the materials and answer questions related to the theme.

**Key-words:** web platform, learning.



# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>7</b>
<b>2</b>	<b>FERRAMENTAS</b>	<b>9</b>
<b>2.1</b>	<b>Framework de Desenvolvimento <i>Web</i></b>	<b>9</b>
2.1.1	<i>Ruby on Rails</i>	9
<b>2.2</b>	<b>Gemas</b>	<b>10</b>
<b>2.3</b>	<b>Editores de Texto</b>	<b>10</b>
2.3.1	Froala Editor	11
2.3.2	TinyMCE	11
<b>2.4</b>	<b>Depuração</b>	<b>11</b>
<b>2.5</b>	<b>Formulários</b>	<b>12</b>
2.5.1	Simple Form	12
2.5.2	Dynamic Form	13
<b>3</b>	<b>DESCRIÇÃO CONCEITUAL</b>	<b>15</b>
<b>3.1</b>	<b>Sistema</b>	<b>15</b>
<b>3.2</b>	<b>Modelagem</b>	<b>16</b>
3.2.1	Escola	16
3.2.2	Projeto	16
3.2.3	Professor	17
3.2.4	Turma	18
3.2.5	Aluno	18
3.2.6	Área do Conhecimento	18
3.2.7	Conteúdo	19
3.2.8	Atividade	19
3.2.9	Material Motivador	19
3.2.10	Exercício	20
3.2.11	Resposta	20
<b>3.3</b>	<b>Casos de Uso</b>	<b>20</b>
<b>4</b>	<b>IMPLEMENTAÇÃO</b>	<b>27</b>
<b>4.1</b>	<b>Peculiaridades</b>	<b>27</b>
<b>4.2</b>	<b>Usuários</b>	<b>27</b>
4.2.1	Acesso	28
4.2.2	Cadastro	28
4.2.3	Perfil	28

4.2.4	Sair do ambiente . . . . .	29
<b>4.3</b>	<b>Projetos . . . . .</b>	<b>29</b>
4.3.1	Métodos Básicos . . . . .	29
4.3.2	Métodos Não Implementados . . . . .	31
<b>4.4</b>	<b>Atividades . . . . .</b>	<b>31</b>
<b>4.5</b>	<b>Materiais Motivadores . . . . .</b>	<b>31</b>
<b>4.6</b>	<b>Exercícios . . . . .</b>	<b>33</b>
<b>4.7</b>	<b>Classes Estáticas . . . . .</b>	<b>34</b>
<b>4.8</b>	<b>Casos de Uso não Implementados . . . . .</b>	<b>34</b>
4.8.1	Pesquisar Projeto . . . . .	34
4.8.2	Criar Dicionário . . . . .	34
4.8.3	Consultar Ajuda . . . . .	35
	<b>CONCLUSÃO . . . . .</b>	<b>37</b>
	<b>Referências . . . . .</b>	<b>39</b>



# 1 Introdução

Uma pesquisa realizada em 2011 pelo Instituto Paulo Montenegro e a ONG Ação Educativa, aponta que no Brasil apenas 25% da população brasileira domina plenamente as habilidades de leitura e escrita(1). Destes, somente 35% das pessoas que concluíram o Ensino Médio e 62% das que concluíram o Ensino Superior são consideradas letradas. O termo “letrado” é utilizado neste texto para designar pessoas capazes de entender e interpretar um texto(2).

O fato do número de jovens que concluem o Ensino Superior e possuem dificuldade na escrita e leitura ser de 75%(1) pode ter origem em uma base educacional deficiente. Desta forma, podemos dizer que crianças que não são devidamente letradas durante os primeiros anos do Ensino Fundamental terão mais dificuldade nos anos subsequentes.

Vários estudos e iniciativas foram desenvolvidos para melhorar o índice de letramento dos estudantes nos primeiros anos de escola. Alguns incluem abordagens em sala de aula(3) e outros envolvem novos métodos como a cibercultura e tecnologia(4).

Dentre as iniciativas que adotam a tecnologia, destacamos o Ideographix, um software que visa auxiliar o letramento através da escrita de textos. Ele foi lançado em 2002 e é resultado de pesquisas realizadas entre 1991 e 2000 na França(5). Este software possui diversas funcionalidades de edição e comparação de textos.

Baseado no Ideographix, Juliana Bueno buscou contribuir no letramento bilíngue efetivo de crianças surdas(6). Ela propôs os conceitos de um aplicativo semelhante ao do software francês sem implementá-lo. Alline de Lara estende o projeto de Bueno para propor um ambiente de autoria para letramento infantil. Este ambiente tem como objetivo melhorar o sistema de leitura e escrita dos alunos. Em contraste com Bueno, de Lara abrange crianças não surdas do Ensino Fundamental. Com o objetivo de implementar o ambiente proposto por de Lara, o trabalho corrente foi proposto.

A presente monografia corresponde a uma implementação do trabalho de Alline, que visa auxiliar professores no processo de letramento de seus alunos em contexto infantil, principalmente, por meio da criação e compartilhamento de materiais base, chamados no sistema de *Materiais Motivadores*, assim como exercícios de fixação relacionados aos materiais propostos. Os alunos terão acesso ao conteúdo e tarefas que estarão dentro de um pacote denominado *Projeto*. O projeto terá como característica principal fomentar a indisciplina para atingir o letramento. Por exemplo, um projeto sobre a fábula dos “Os Três Porquinhos” pode conter materiais que estejam ligados aos conteúdos de gramática, matemática e ciências sem ferir o propósito educacional da história infantil. A hierarquia e componentes do sistema serão descritos no capítulo 3.

Neste texto é utilizado o termo “software de autoria” com o objetivo de descrever sistemas que fornecem mecanismos para criação de conteúdo dentro de si, através de imagens, vídeos, sons, tabelas, textos, entre outros (7). No caso do sistema apresentado nesta monografia, professores são capazes de criar materiais base e exercícios mesmo sem conhecimento de programação, enquanto os alunos criam respostas baseadas nos documentos feitos pelo professor.

A fim de obter maior facilidade na utilização do sistema para pontos geograficamente distantes do Brasil, além de não precisar de instalação, o projeto foi desenvolvido para resultar em um sistema web pois tais características são presentes neste tipo de plataforma. A escolha do framework de desenvolvimento Ruby on Rails é justificada em função da rápida prototipação e da facilidade de entendimento do Ruby on Rails, detalhado no capítulo 2 deste trabalho.

## 2 Ferramentas

Este capítulo apresenta as ferramentas utilizadas para o desenvolvimento do sistema de letramento online apresentado nesta monografia, o **SISLETRON**. Em particular, na seção 2.1 há uma descrição mais detalhada do framework Ruby on Rails, capaz de facilitar a criação de sites que utilizam banco de dados. Esta linguagem possui diversas bibliotecas de código disponíveis na web que são chamadas de *gemas*, responsáveis por facilitar a produção e depuração dos sistemas.

### 2.1 Framework de Desenvolvimento *Web*

Um framework de aplicação web é um software capaz de dar suporte ao desenvolvimento de websites dinâmicos, aplicações web, serviços web e recursos web(8). O objetivo de um framework é de aliviar a carga de trabalho associada a atividades comuns de desenvolvimento de um sistema web. Por exemplo, muitos frameworks disponibilizam bibliotecas para acesso ao banco de dados e até reuso de código.

#### 2.1.1 *Ruby on Rails*

Ruby é uma linguagem de programação orientada a objetos moderna e fácil de escrever e ler. Esta linguagem facilita a criação de métodos que parecem ser extensões da sintaxe. Ruby possui o conceito *DRY* que significa *Don't Repeat Yourself*, ou seja, “não se repita”(9) que sugere que as partes do sistema devem ser construídas em um único e respectivo lugar.

Rails é uma biblioteca Ruby para o desenvolvimento de aplicações Web. Sua versão inicial foi lançada em dezembro de 2005. Este framework utiliza padrões do Ruby para seu funcionamento, além explorar a arquitetura do *MVC*. Esta sigla, por sua vez, significa *model*, *view* e *controller*, responsáveis pela modularização e reuso do código, além de possibilitar múltiplas interfaces de serem aplicadas(8).

Na prática, Rails separa o sistema nas três segmentações descritas anteriormente. A estruturação do banco de dados é feita nos modelos, responsáveis por manter o estado da aplicação. A parte visual das páginas é descrita nas *views*, que geram a interface com o usuário e são geralmente baseadas no modelo. Requisições e métodos são tratados pelo controlador, responsável na maior parte dos casos por receber dados gerados pelo usuário. O controlador recebe eventos, interage com o modelo e dispõe dados de maneira organizada através das *views*. Deste modo, o programador é capaz não apenas de manter um código enxuto mas de mantê-lo legível para outros programadores.

Em uma aplicação Rails, as requisições são primeiramente enviadas para uma rota, que tem o papel de analisá-las e encaminhá-las para o destino correto. Em alguns casos a rota também é responsável por preparar informação para a *view*, que a traduz em formato visual a interface para o usuário.

Outro aspecto positivo de Rails é seu método de desenvolvimento ágil no qual foi baseado. Isto significa que o desenvolvimento é focado nos conceitos que sugerem escolher(9):

- Indivíduos e interações a processos e ferramentas
- Software que funciona a documentação compreensiva
- Colaboração com clientes a negociação de contrato
- Responder às mudanças a seguir o plano

Desta forma, a escolha de Ruby on Rails para o desenvolvimento do sistema proposto é justificada, além de outros fatores, pela possibilidade de criação de trabalhos futuros na linha desta monografia, facilmente adaptável a novos colaboradores devido à baixa dificuldade de entendimento e de manutenção do código.

## 2.2 Gemas

Gemas são pacotes de códigos – assim como bibliotecas nas linguagens estruturais (C, por exemplo) – que também auxiliam no rápido desenvolvimento. Diversas gemas para diferentes objetivos estão espalhadas pela web. Este trabalho utilizou algumas delas para o funcionamento correto da aplicação que são descritas nas seções a seguir.

O framework Rails também pode ser considerado uma gema. Entretanto, ele mesmo utiliza várias outras gemas como por exemplo a chamada *ActiveRecord* que é responsável pela abstração do banco de dados. Neste trabalho são denominadas de “gemas padrão” aquelas que são necessárias para o funcionamento do Rails.

Nas próximas seções são apresentadas as funcionalidades desejadas para a construção do SISLETRON, assim como gemas que foram incorporadas para a implementação do trabalho descrito nesta monografia. O processo de seleção destas gemas baseou-se em buscas no site *RubyGems* (repositório padrão de gemas) e teste de funcionalidades.

## 2.3 Editores de Texto

Editores de texto são softwares que possibilitam a formatação de um texto de acordo com o desejo ou a necessidade do usuário. É possível separar esses softwares em

duas categorias: *WYSIWYG* e de texto plano. A primeira significa *what you see is what you get* que se traduz em “aquilo que vê é aquilo que recebe”. Exemplos de softwares que seguem essa linha são LibreOffice e Microsoft Word. O segundo representa editores de texto que não possuem mudança gráfica no resultado final, geralmente usados para produção de códigos computacionais, como o Gedit, VIM e Wordpad (Bloco de Notas). Apenas a primeira categoria será tratada neste trabalho, pois os usuários – no caso alunos do ensino fundamental e professores – podem utilizar funções como negrito, itálico e cores nos textos que escreverão.

### 2.3.1 Froala Editor

Froala Editor é um editor de texto WYSIWYG HTML, atualmente na versão 2.0. Sua interface é elegante tornando o uso intuitivo pelo usuário. Além do design não há grande diferença com o editor de texto TinyMCE descrito na subseção 2.3.2. Entretanto, a licença de uso do Froala Editor é paga. Como a característica do sistema desenvolvido nesta monografia é de ser software aberto e livre, optou-se pelo uso do editor de texto seguinte.

### 2.3.2 TinyMCE

TinyMCE é uma plataforma de edição de texto baseado em JavaScript para sistemas web compatível com Ruby on Rails que possibilita a implementação de editores de texto WYSIWYG. A versão usada para construir o SISLETRON foi o TinyMCE Community 4.2.5.

Esta plataforma de código aberto é altamente modificável e possui plugins próprios, gerenciados no momento de declaração do script no documento HTML. Plugins geralmente são dispostos na barra de ferramentas do aplicativo, responsáveis pela formatação do texto, como imprimir, quebrar linha, quebrar página, contador de palavras, entre outros.

TinyMCE é compatível com múltiplos navegadores, incluindo Internet Explorer, Mozilla Firefox, Safari, Opera e Google Chrome em diferentes sistemas operacionais. Também possui suporte a mais de 40 línguas, incluindo Português Brasileiro.

Em particular, o TinyMCE foi utilizado neste trabalho principalmente na criação de materiais motivadores, como descrito no capítulo 4.

## 2.4 Depuração

No desenvolvimento de sistemas em geral, principalmente em curto prazo, é essencial obter meios de depuração de código que facilitem o trabalho do programador. A gema *Better Errors* (versão 2.1.1) substitui a página padrão de erro do Ruby on Rails para uma

página mais interessante do ponto de vista de tratamento de erros. Ela possui as seguintes funcionalidades que são interessantes de serem mencionadas:

- Rastro completo do erro
- Inspeção de código fonte para toda a pilha com demarcação de texto
- Inspeção local e de instâncias de variáveis
- Console em tempo real do sistema no momento do erro

A gema adicional *binding\_of\_caller* foi adicionada para possibilitar o funcionamento do console.

Em particular, a gema *Better Errors* foi muito útil no desenvolvimento do sistema devido a possibilidade de obter resposta do banco de dados ao utilizar métodos dos modelos. Assim, além de exibir o erro de forma mais completa, também auxiliou na redução do tempo de desenvolvimento pois fornece o console para testes em tempo real, descartando a necessidade de checagem no console do rails em um terminal linux.

É também importante mencionar que esta gema deve ser utilizada **somente** na configuração de desenvolvimento. Ao ser utilizada na configuração de produção, esta gema possibilita os usuários de obterem informações sigilosas do sistema, causando um grande problema de segurança. Para isso, foi utilizada uma verificação no código para habilitar a gema apenas quando na configuração de desenvolvimento.

## 2.5 Formulários

Ruby on Rails possui *helpers* – métodos globais que facilitam o desenvolvimento – nativos para a utilização de formulários nas páginas. Eles são colocados nas views para servir de interface entre o usuário e os controladores do sistema. Infelizmente, o formulário padrão do Rails carece de alguns fatores que foram importantes para o desenvolvimento do SISLETRON. Nas subseções a seguir estão listadas gemas que tiveram seu uso necessário para o comportamento adequado da interface.

### 2.5.1 Simple Form

*Simple Form* (versão 3.1) é uma gema que visa tornar a criação de componentes para os formulários mais flexível. Ou seja, na prática auxilia na definição do layout dos formulários, que tinham seu modelo visual restrito pelo *helper* nativo de Rails. Desta forma, classes de CSS podem ser imbutidas de forma separada para cada campo de entrada do formulário de acordo com a necessidade do sistema, além de outras funcionalidades também interessantes, mas que não foram utilizadas neste trabalho.

Outra diferença em relação ao formulário nativo é a simplicidade na chamada dos métodos para a criação dos formulários, que auxiliam na clareza do código e na redução do tempo de desenvolvimento.

### 2.5.2 Dynamic Form

*Dynamic Form* é uma gema simples que adiciona alguns métodos *helper* com o objetivo de auxiliar modelos de formulário do Rails 3. Neste trabalho, surgiu a necessidade de utilizá-la para implantar mensagens de erro nos construtores dos formulários, cujas posições permaneciam fixas em local indesejado ao utilizar o método nativo de Rails.





## 3 Descrição Conceitual

Neste capítulo, apresenta-se o sistema derivado dos modelos conceituais. A seção 3.1 é responsável por detalhar o funcionamento do sistema no nível de usuário. A seção 3.2 descreve como os dados foram modelados no sistema, incluindo o diagrama de classe.

### 3.1 Sistema

A especificação do sistema é dada da seguinte forma: um professor cria um projeto para uma ou mais turmas de alunos de uma escola. O professor que cria o projeto e aquele que executa, ou seja, que põe em prática, pode ser diferente. Cada conteúdo pertence a uma área de conhecimento. O projeto consiste de diferentes atividades que são compostas de materiais motivadores e suas tarefas.

Para ilustrar o sistema, o projeto exemplo da fábula Os Três Porquinhos é descrito a seguir:

**Escola:** Lápis de Cor

**Projeto:** Os Três Porquinhos

**Professor criador:** José da Silva, Juliano Andrade

**Professor executor:** Juliano Andrade, Amélia Pinheiro

**Turma:** 1B

**Alunos:** Ana Carolina Louro, André Souza, Diego Trindade, ..., Zacarias Machado.

**Área do Conhecimento:** Ciências, Matemática

**Conteúdo:** Reino Animal, Números Naturais

**Atividade:** Aprendendo com os Porquinhos

**Material Motivador:** Os três porquinhos são animais que vivem felizes no campo. Eles possuem narizes grandes, coloração rosa e patas curtas. Na floresta ali perto mora seu predador natural. O nome dele é Lobo Mau e adora caçar porquinhos. Cada porquinho possui uma casa diferente. O primeiro porquinho tem uma casa de palha que aguenta três sopros do lobo. O segundo porquinho possui uma casa de madeira que aguenta cinco sopros. O terceiro e último porquinho possui uma casa de tijolo que aguenta nove sopros do terrível Lobo Mau (...).

**Exercício:** 1) Quais animais são descritos no texto e qual suas características? 2) Quantas vezes o Lobo Mau deverá soprar para derrubar as casas dos três porquinhos?

O exemplo do projeto “Os Três Porquinhos” demonstra que nas famosas fábulas contadas para crianças no início do ensino há diferentes conteúdos que podem ser tratados. Neste sentido, os projetos reforçarão o letramento de forma expandida em termos de

abrangência curricular através do uso do sistema, responsável por exibir quais conteúdos estão sendo tratados. Entretanto, o material motivador e os exercícios não mencionarão qual área do conhecimento estão envolvidos na questão específica, focando no aspecto interdisciplinar do letramento.

## 3.2 Modelagem

Nesta seção o sistema será detalhado de forma a exibir seus modelos conceituais. Os componentes (projetos, escola, turmas, etc) serão chamados de classes e suas propriedades serão chamadas de atributos. As subções aqui contidas descrevem os atributos das classes e as relações entre os componentes entre si, fundamentais para o funcionamento correto do sistema, pois facilitam a comunicação entre as partes.

### 3.2.1 Escola

No sistema, a escola representa a instituição de ensino em que as turmas estão contidas. Por definição, um projeto pode ser aplicado em muitas turmas. No desenvolvimento atual não há restrição por escolas, ou seja, um projeto pode conter turmas de diferentes escolas.

Os atributos da classe Escola são: id próprio, nome e telefone. O primeiro ocorre obrigatoriamente em todas as classes devido à estrutura de Ruby on Rails e será ocultado neste trabalho daqui em diante, exceto quando necessário. Os atributos nome e telefone foram considerados suficientes para caracterizar a escola no sistema.

A escola possui um relacionamento de uma para muitas turmas apenas. Durante o processo de criação do modelo conceitual do sistema, foi decidido que a classe escola não possui importância significativa no contexto deste trabalho exceto para separação dos alunos.

### 3.2.2 Projeto

O projeto é a classe mais importante do sistema e a que possui mais conexões com outros componentes. Como foi possível perceber através do exemplo citado na seção 3.1, o projeto contém diversas informações que estão separadas em outras classes.

Os atributos da classe Projeto são: título, descrição e avaliação. Título e descrição são fundamentais para o entendimento do projeto a nível de usuário. Avaliação, por sua vez, refere-se ao resultado do projeto após a finalização do mesmo, e será atribuída por um dos professores que o executou.

O Projeto foi dividido em duas partes: plano e execução. Um professor ao criar um projeto não necessariamente fará parte da equipe que o colocará em prática. A esta

relação entre professores que fazem o planejamento de projetos e respectivos projetos, denominou-se plano. Execução por sua vez refere-se à relação dos projetos e dos professores que os executaram.

Os demais relacionamentos da classe Projeto são:

- Um projeto pode ser aplicado em mais de uma turma;
- Uma turma pode ter mais de um projeto;
- Um projeto possui várias atividades;
- Um projeto contém vários conteúdos;
- Um conteúdo pode estar em mais de um projeto.

Desta forma, através de uma instância da classe Projeto é possível obter as turmas, conteúdos, atividades e professores que estão relacionados com o projeto.

### 3.2.3 Professor

O professor é responsável por criar ou aplicar os projetos nas turmas de alunos. No modelo feito para este trabalho, o professor não possui distinção de cargo ou papel, exceto quando está nas relações Execução e/ou Plano. Por isso, no sistema geral um professor não possui permissões além de um outro professor.

Os atributos da classe Professor são: nome, data de nascimento, identificador para login e senha. Como o objetivo do trabalho era a construção do sistema com ênfase no funcionamento de suas funcionalidades, o e-mail como parte do login não foi considerado um bom atributo e consequentemente deixado de lado. Caso o cadastro de professores seja livre em trabalhos futuros, recomenda-se a substituição do atributo identificador para o e-mail, com suas respectivas funções de checagem e envio de e-mail para recuperação de senha, entre outros.

Os relacionamentos da classe Professor são:

- Um professor pode lecionar em mais de uma turma;
- Uma turma pode ter mais de um professor;
- Um professor pode criar mais de um projeto;
- Um projeto pode ser planejado por mais de um professor;
- Um professor pode executar mais de um projeto;
- Um projeto pode ser executado por mais de um professor.

### 3.2.4 Turma

A classe Turma contém os atributos id da escola, ano, turma e descrição. Cada turma está relacionada a apenas uma escola. Portanto, o id da escola torna-se necessário para se obter em qual escola os alunos estão matriculados. O ano e a turma referem-se às propriedades de uma classe escolar, como no exemplo citado na seção 3.1, principalmente voltado ao ensino fundamental. Entretanto, para a utilização do SISLETRON também no Ensino Médio, além de contemplar outras possibilidades, foi percebida a necessidade de criação do atributo descrição, que deve ser usado de forma a diferenciar turmas cujos ano e turma não são suficientes para fazê-lo.

Os relacionamentos da classe Turma são:

- Uma turma pertence a uma escola;
- Uma turma pode ter mais de um professor;
- Um professor pode lecionar em mais de uma turma;
- Uma turma pode ter mais de um projeto;
- Um projeto pode ser executado em mais de uma turma;
- Uma turma tem mais de um aluno.

### 3.2.5 Aluno

Aluno é a única classe de usuário do sistema além do Professor. Seus atributos não diferem do outro usuário: nome, data de nascimento, identificador de login e senha. Sua responsabilidade como atuante no sistema é ler os materiais motivadores e responder seus exercícios.

Os relacionamentos da classe Aluno são apenas: pertence a uma turma e possui muitas respostas. O acesso ao projeto é permitido automaticamente quando há relação entre a turma do aluno e um projeto.

### 3.2.6 Área do Conhecimento

Neste trabalho, área do conhecimento se refere às matérias letivas do ensino fundamental. Entre elas, estão Matemática, Português, Ciências, História, Geografia, Artes e Educação Física. Cada área de conhecimento contém diversos conteúdos. Voltando ao exemplo da seção 3.1 para demonstração do caso, o conteúdo Reino Animal pertence à área do conhecimento Ciências. Como atributos a classe possui apenas título. Sua relação é apenas de uma para muitos conteúdos.

### 3.2.7 Conteúdo

Como mencionado na subseção anterior, um conteúdo pertence a uma área do conhecimento. Os conteúdos são usados para ampliar a descrição dos projetos e, em trabalhos futuros, serão usados como palavras-chave para busca no sistema.

Os atributos da classe Conteúdo são: id da área de conhecimento, título, descrição e ano. A descrição serve para detalhar os conteúdos que possuam alguma peculiaridade ou necessitem de explicação além do título. O atributo ano é usado principalmente para separação dos conteúdos pelos anos em que são distribuídos pela grade escolar durante o ensino fundamental, baseados em uma especificação arbitrária, como por exemplo do MEC ou da Prefeitura de Curitiba.

Os relacionamentos da classe Conteúdo são de muitos para uma área de conhecimento, além de muitos conteúdos para muitos projetos. Alternativamente, esta última relação pode ser feita com materiais motivadores ao invés de projeto, porém não foi usada pois futuramente projetos podem ser pesquisados através da direta ligação com os conteúdos.

### 3.2.8 Atividade

Para melhor entendimento, a atividade pode ser relacionada com uma unidade de um livro educacional. A unidade possui apenas um título para referência. Por sua vez, o capítulo dentro da unidade possui textos e imagens que auxiliam o entendimento do conteúdo pelo aluno e exercícios que permitem a fixação do conteúdo através da prática. Neste sentido, um livro (projeto) possui muitas unidades (atividades) que por sua vez possuem muitos capítulos (materiais motivadores) e que possuem muitos exercícios.

Voltando ao sistema SISLETRON, o modelo conceitual da Atividade possui apenas título como atributo próprio e o id do projeto. Sua relação é de pertencer a um único projeto e possuir vários materiais motivadores.

### 3.2.9 Material Motivador

Os materiais motivadores são responsáveis por dar o embasamento necessário aos alunos para que possam resolver os exercícios. Assim, os atributos da classe Material Motivador são: id da atividade, título, conteúdo e tipo. O título do material motivador deve deixar claro o que será tratado dentro dele. O conteúdo é responsável por armazenar toda a informação em formato de *string*, como detalhado no capítulo 4. O tipo é usado para distinguir os materiais em seu formato multimídia, ou seja, os usuários poderão, em trabalhos futuros, filtrar materiais de texto, vídeo, áudio ou todos (multimídia), de acordo com a especificação feita na hora da criação do material.

O relacionamento com outras classes é apenas de uma atividade para muitos materiais motivadores, que por sua vez possuem muitos exercícios.

### 3.2.10 Exercício

Os exercícios devem sempre estar atrelados a um material motivador que sirva de base, fornecendo conteúdo suficiente para que os alunos consigam completá-lo. Os atributos da classe Exercício são: id do material motivador, enunciado, correção, tipo e nota. Trabalhos futuros com diferentes tipos de exercício necessitarão de outros atributos, possuindo talvez duas partes para o enunciado. O atributo correção serve para o sistema automaticamente retornar uma nota à resposta do aluno baseado no conteúdo do campo para determinados tipos de exercício, embora esta seja uma funcionalidade não realizada neste trabalho. Os tipos de exercício seriam determinados pelo professor ao criá-lo. Alguns dos tipos já foram pensados mas ainda não implementados neste trabalho, como por exemplo: múltipla escolha, dissertação, preenchimento de lacunas, entre outros.

A relação dos exercícios com outras classes é de um material motivador para muitos exercícios, que por sua vez possuem muitas respostas de diferentes alunos.

### 3.2.11 Resposta

As respostas são relacionadas aos exercícios. Cada aluno pode possuir apenas uma resposta para um determinado exercício. O atributo conteúdo refere-se ao corpo da resposta criada pelo aluno para algum exercício. Os outros atributos incluem o id do aluno e o id do exercício, sendo essa a relação correspondente.

## 3.3 Casos de Uso

Um diagrama de caso de uso tem como objetivo descrever os requisitos do ambiente para cada cenário possível, tornando explícita a sequência de passos para a interação dos usuários com o ambiente (10). Os casos de uso são compostos por atores, casos de uso em si e relacionamento entre ambos os anteriores. Desta forma, é possível visualizar o comportamento do sistema através do ponto de vista do ator.

Em conjunto com De Lara, foram criados os cenários do ambiente através dos casos de uso, como mostra a figura 1.

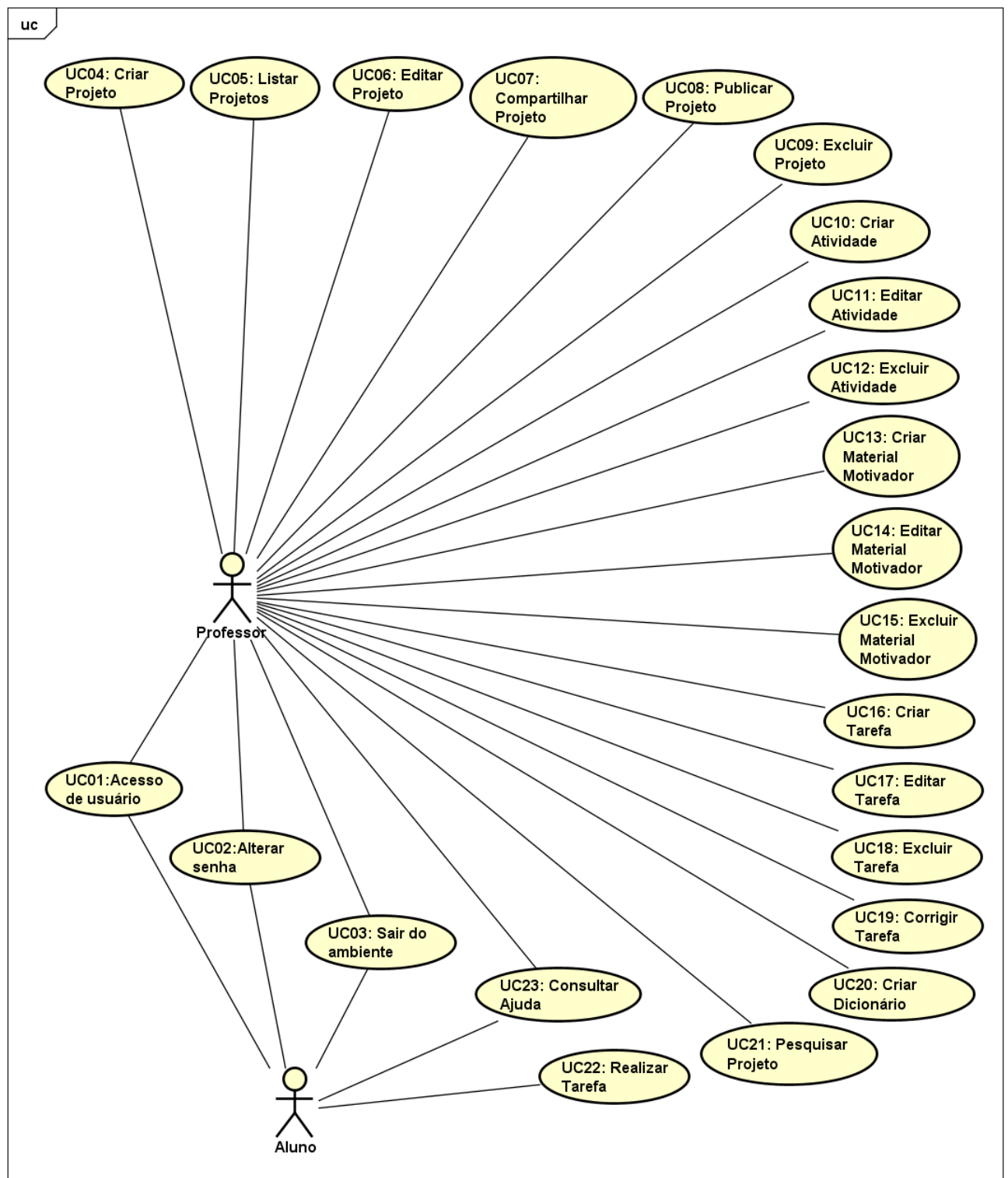


Figura 1: Diagrama de Caso de Uso

A seguir estão listados os casos de uso apresentados na figura 1.

**UC01:Acesso de usuário** este caso de uso pode ser iniciado pelo professor ou pelo aluno. O seu objetivo é fornecer o acesso ao ambiente proposto;

**UC02:Alterar senha** este caso de uso é iniciado pelo professor ou pelo aluno. Com este caso de uso é possível alterar a senha de acesso ao ambiente;

**UC03: Sair do ambiente** este caso de uso é iniciado pelo professor ou pelo aluno. O caso de uso permite que o usuário saia do ambiente;

**UC04: Criar Projeto** este caso de uso é iniciado pelo professor. Com este caso de uso é possível criar projetos de estudo para serem trabalhados com os alunos;

**UC05: Listar Projetos** este caso de uso é iniciado pelo professor. Através deste caso de uso é possível listar os projetos associados ao professor;

**UC06: Editar Projeto** este caso de uso é iniciado pelo professor. Com este caso de uso o professor pode editar os projetos relacionados a ele;

**UC07: Compartilhar Projeto** este caso de uso é iniciado pelo professor. O caso de uso permite que o professor compartilhe o projeto associado a ele com outros professores que trabalhem na mesma escola que o professor;

**UC08: Publicar Projeto** este caso de uso é iniciado pelo professor. Este caso de uso é utilizado para publicar os projetos já aplicados e avaliados pelo professor;

**UC09: Excluir Projeto** este caso de uso é iniciado pelo professor. O caso de uso permite que o professor exclua os projetos relacionados a ele;

**UC10: Criar Atividade** este caso de uso é iniciado pelo professor. Com este caso de uso é possível criar atividades no ambiente;

**UC11: Editar Atividade** este caso de uso é iniciado pelo professor. Através deste caso de uso é possível editar as atividades do ambiente proposto;

**UC12: Excluir Atividade** este caso de uso é iniciado pelo professor. O caso de uso permite a exclusão das atividades no ambiente proposto;

**UC13: Criar Material Motivador** este caso de uso é iniciado pelo professor. O caso de uso possibilita a criação de materiais motivadores, sendo estes materiais compostos por: textos, imagens, vídeos, documentos e *links*.

**UC14: Editar Material Motivador** este caso de uso é iniciado pelo professor. Através deste caso de uso é possível editar os materiais motivadores cadastrados no ambiente.



**UC15: Excluir Material Motivador** este caso de uso é iniciado pelo professor. Com este caso de uso o professor pode excluir os materiais motivadores cadastrados no ambiente.

**UC16: Criar Tarefa** este caso de uso é iniciado pelo professor. O caso de uso permite que o professor crie tarefas para disponibilizar para os alunos.

**UC17: Editar Tarefa** este caso de uso é iniciado pelo professor. Através deste caso de uso o professor pode editar as tarefas.

**UC18: Excluir Tarefa** este caso de uso é iniciado pelo professor. Através deste caso de uso o professor pode excluir as tarefas cadastradas no ambiente.

**UC19: Corrigir Tarefa** este caso de uso é iniciado pelo professor. O caso de uso possibilita a correção das tarefas realizadas pelos alunos.

**UC20: Criar Dicionário** este caso de uso é iniciado pelo professor. Através este caso de uso é possível criar dicionários com as palavras encontradas nos materiais motivadores, podendo o professor filtrar as palavras informando um sufixo ou prefixo específico.

**UC21: Pesquisar Projeto** este caso de uso é iniciado pelo professor. Através deste caso de uso o professor pode pesquisar os projetos publicados no ambiente.

**UC22: Realizar Tarefa** este caso de uso é iniciado pelo aluno. Com este caso de uso o ambiente permite que o aluno realize as tarefas disponibilizadas pelo professor.

**UC23: Consultar Ajuda** este caso de uso é iniciado pelo professor ou pelo aluno. Este caso de uso permite que o usuário consulte o texto de ajuda presente no ambiente.

Cada caso de uso pode ser detalhado através do que é chamado de “Descrição de Caso de Uso”. Apresentar todas as descrições de casos de uso foge do escopo deste trabalho. Entretanto, para exemplificar a descrição de um caso de uso do SISLETRON foi escolhido o UC04: Criar Projeto, apresentado abaixo.

#### **UC04: Criar Projeto**

**Descrição** Este caso de uso é iniciado pelo professor quando for necessário:

1. Criar um projeto de estudos.

**Pré-condições** Este caso de uso pode iniciar somente se:

1. O ambiente tiver executado o UC01 – Acesso de usuário;
2. O professor é quem está autenticado no ambiente;

**Pós-condições** Após o final normal deste caso de uso o ambiente deve:

1. Salvar o projeto criado pelo professor.

**Ator Primário** Professor

**Fluxos de Eventos** Principais

1. O ambiente apresenta a tela Inicial do Ambiente;
2. O professor clica no menu “Projeto”;
3. O ambiente apresenta os submenus “Criar Projeto” e “Listar Projetos”;
4. O professor clica no submenu “Criar Projeto”;
5. O ambiente abre a tela Criar Projeto;
6. O professor informa os campos: o Nome e Descrição associados ao projeto;
7. O professor seleciona uma área do conhecimento no campo Área do Conhecimento;
8. O ambiente apresenta uma lista dos conteúdos que pertencem a área do conhecimento escolhida pelo professor;
9. O professor marca os conteúdos que deseja associar ao projeto e clica no botão “Incluir Conteúdos”; **A1. E1.**
10. O ambiente adiciona os conteúdos marcados pelo professor na lista de conteúdos associados ao projeto;
11. O professor seleciona uma escola no campo Escola;
12. O ambiente apresenta uma lista das turmas que o professor leciona na escola selecionada;
13. O professor marca as turmas que deseja associar ao projeto e clica no botão “Incluir Turmas”; **E2.**
14. O ambiente adiciona as turmas marcadas pelo professor na lista de turmas associados ao projeto;
15. O professor clica no botão “Salvar Projeto”; **E3. E4. E5.**
16. O ambiente salva o novo projeto, redireciona o professor para a tela do projeto e mostra a mensagem “Projeto criado com sucesso”.
17. O caso de uso é finalizado.

**Fluxos de Eventos** Alternativos

**A1.** O professor não está procurando nenhum dos 10 conteúdos exibidos na tela.

1. O professor clica em uma opção da paginação;
2. O ambiente exibe outros conteúdos relacionados a área do conhecimento;

3. O caso de uso retorna ao Fluxo Principal 13.

### **Fluxos de Excessões**

**E1.** O professor não marcou nenhum conteúdo para ser incluído.

1. O ambiente verifica que o professor não marcou nenhum conteúdo, retorna a mensagem “Por favor, marque os conteúdos a serem associados ao projeto.” e destaca o campo a coluna com as caixas de seleção da tabela Área de conhecimento e Conteúdos de vermelho e negrito;
2. O caso de uso retorna para o Fluxo Principal 9.

**E2.** O professor não marcou nenhuma turma para ser incluída.

1. O ambiente verifica que o professor não marcou nenhuma turma, retorna a mensagem “Por favor, marque as turmas a serem associados ao projeto.” e destaca o campo a coluna com as caixas de seleção da tabela Escola e Turma de vermelho e negrito;
2. O caso de uso retorna para o Fluxo Principal 13;

**E3.** O campo Nome está em branco.

1. O ambiente verifica que o campo Nome não foi preenchido, retorna a mensagem “Por favor, informe o campo Nome para cadastrar o novo projeto” e destaca o campo Nome de vermelho e negrito;
2. O caso de uso retorna para o Fluxo Principal 6.

**E4.** O campo Descrição está em branco.

1. O ambiente verifica que o campo Descrição não foi preenchido, retorna a mensagem “Por favor, informe o campo Descrição para cadastrar o novo projeto.” e destaca o campo Descrição de negrito e com a cor vermelha;
2. O caso de uso retorna para o Fluxo Principal 6.

**E5.** A lista de conteúdos associados ao projeto está em branco.

1. O ambiente verifica que a lista de conteúdos associados ao projeto está em branco, retorna a mensagem “Por favor, associe um conteúdo ao projeto para cadastrar o novo projeto.” e destaca a sessão Conteúdo com a cor vermelha
2. O caso de uso retorna para o Fluxo Principal 7.

Embora tenham sido elaborados vinte e três casos de uso, este trabalho teve como objetivo implementar um conjunto seletivo dos apresentados nesta seção. Os detalhes da implementação podem ser vistos no capítulo [4](#).

## 4 Implementação

Neste capítulo será descrito como foi feita a implementação do SISLETRON. As seções descrevem as diferenças de acesso para professores e alunos em determinado contexto do sistema, comentam dificuldades ou eventuais problemas encontrados durante a implementação e a seção 4.8 lista os casos de uso apresentados no capítulo anterior que não foram implementados neste trabalho.

### 4.1 Peculiaridades

Algumas peculiaridades são encontradas por todo o sistema e devem ser mencionadas previamente às outras seções. Primeiramente, para controladores e rotas foi usado o padrão do Ruby on Rails de pluralização de palavras, ou seja, palavras que não são encontradas no dicionário nativo do framework em inglês são pluralizadas adicionando a letra S ao final. Assim, palavras em português que terminam em R ou L, por exemplo, são pluralizadas da mesma forma, gerando controladores como “material\_motivadores” e “professors”, por exemplo. Outro ponto que deve ser mencionado é a inexistência de métodos de verificação das entradas do sistema, havendo algumas exceções. Por exemplo, é possível cadastrar um aluno cuja data de nascimento é escrita por extenso ou contendo um grande número de caracteres. Uma das exceções é a validação na presença e na quantidade de caracteres do nome do usuário. Essas validações podem ser editadas ou adicionadas nos modelos de cada classe.

Estes e outros problemas encontrados no sistema possuem solução, mas devido ao curto período de tempo de confecção desta monografia, o autor preferiu deixá-las de lado para dar prioridade a outras partes do sistema.

### 4.2 Usuários

Nesta seção serão apresentados os detalhes de implementação das classes Professor e Aluno e suas funcionalidades.

Os usuários são a parte fundamental do sistema, pois sem eles não há construção de materiais e interação através de leituras e respostas.

Como apresentado no capítulo 3, há apenas dois tipos de usuário: professor e aluno. As subseções a seguir descrevem os detalhes da implementação do cadastro, acesso e do perfil dos usuários.

### 4.2.1 Acesso

O sistema foi feito de forma a permitir o acesso às informações somente através do login dos usuários no sistema, ou seja, usuários não registrados não tem permissão de visualização dos conteúdos do sistema, exceto pela página inicial, de cadastro e login.

Esta forma de impedimento foi realizada a partir do método *authorize* descrito no *application controller*. A partir do método *helper* “*before\_action :authorize*” chamado em cada controlador de interesse, o método checa se existe um usuário com seção ativa no sistema. Caso contrário, retorna à página inicial e impede o acesso ao conteúdo.

A implementação do acesso está no controlador *sessions*, responsável por criar a seção de login no sistema. Nele, as informações de entrada necessárias ao acesso do usuário são “identificador” e “senha”, ambos atributos das classes Aluno e Professor.

### 4.2.2 Cadastro

O sistema implementado neste trabalho possui cadastro livre aos usuários devido à facilidade de testes. Embora exista possibilidade de integração com bancos de dados de escolas municipais em trabalhos futuros, esta opção foi descartada para a primeira versão do sistema.

A implementação do cadastro ocorre em controladores separados para cada tipo de usuário. Desta forma, duas rotas foram feitas: *cadastro\_professors* e *cadastro\_alunos*, cujos links de acesso permanecem no topo da página inicial, ao lado do login.

O formulário de cadastro está nas páginas “alunos/new” e “professors/new”. Os campos exigidos pelo modelo são *identificador*, *nome* e *password*. Além destes, o formulário também contém *data de nascimento*.

### 4.2.3 Perfil

O usuário pode acessar seu perfil clicando em seu nome localizado no canto superior direito da página. Apenas os próprios usuários conseguem ver sua página de perfil, não havendo permissão de acesso em perfis alheios.

Os alunos ao clicarem no link de visualização do perfil serão redirecionados à página correspondente, que mostra seus dados cadastrais. Os professores, entretanto, são redirecionados a uma página que contém as turmas em que leciona e os projetos, além dos dados cadastrais.

Em ambos os casos existe o botão para editar cadastro ao final da página, que redireciona à página correspondente.

#### 4.2.4 Sair do ambiente

A funcionalidade de sair do ambiente é feita através do link que fica no topo da página ao lado do nome do usuário. O link chama o método *destroy* no controlador *sessions*, adicionando um valor nulo à variável de sessão do usuário. Desta forma, o usuário perde as permissões para visualizar as páginas do sistema.

### 4.3 Projetos

Nesta seção serão apresentados os detalhes de implementação da classe Projeto e suas funcionalidades.

Os projetos foram implementados de forma a conter todas as suas atividades, materiais motivadores, exercícios e respostas. Neste sentido, para facilitar a navegação entre os componentes de um projeto, foi implementado um *breadcrumb* que é mostrado como primeira informação abaixo do topo da página. *Breadcrumb* significa navegação estrutural em português, e na prática corresponde ao caminho feito pelo usuário para chegar a determinado ponto no sistema. Para este trabalho a estrutura é feita da seguinte forma: Projeto → Atividade → Material Motivador → Exercício.

#### 4.3.1 Métodos Básicos

Nesta subseção serão descritas as implementações dos métodos básicos *CRUD* (Criar, Listar, Editar e Deletar).

Um projeto pode ser criado apenas por professores. Esta restrição foi implementada através do método *helper* “*before\_action :require\_professor, only: [:create, :new, :edit, :update]*”. O método *require\_professor* foi declarado no controlador da aplicação e sua função é de retornar à página inicial caso o usuário seja um aluno ou não esteja com variável de sessão ativa. Na página de projetos o professor pode criá-lo clicando no botão Criar Projeto. O formulário de criação de projetos contém os seguintes atributos: título, descrição, conteúdos e turmas. Na versão do sistema feita para esta monografia, não foi implementada a opção de atribuição de professores para planejamento ou execução dos projetos. Também, não é possível atribuir um valor para a avaliação do projeto nesta versão do sistema. Após a criação, o usuário é redirecionado à página de lista de projetos.

A lista de projetos é a página inicial (*index*) dos projetos. Para o professor, esta lista separa os projetos em duas partes: os projetos que foram criados pelo professor e os projetos em que ele está executando, representados, respectivamente, pelos textos “projetos que criou” e “seus projetos em andamento”. Para o aluno, só a última lista é exibida. Ao clicar no nome do projeto, o usuário é redirecionado a página de exibição do projeto correspondente.



Figura 2: Captura de tela do sistema: tela exibição do projeto para professores

Na página de exibição do projeto desejado são exibidos seu título, avaliação (se houver), descrição, lista de conteúdos, lista de links para atividades, lista de professores que aplicam, lista de professores que planejam e lista de turmas em que o projeto está sendo aplicado. Para os professores, é exibido um botão para criação de atividades e um botão de edição do projeto.

A edição do projeto é feita apenas pelos professores responsáveis pelo projeto, ou seja, todos aqueles que o executam ou planejam. A seguinte mensagem é exibida caso essa restrição tente ser violada: “Você não tem autorização para editar este projeto”. O formulário de edição de projetos é idêntico ao do formulário de criação.

O método de exclusão de projetos não foi implementado. Este método seria possível após declarar a dependência destrutiva nos modelos de atividade, material motivador, exercício e respostas. Desta forma, caso um projeto seja excluído, todos os seus componentes também seriam destruídos.



### 4.3.2 Métodos Não Implementados

Nesta subseção serão explicados os métodos Compartilhar e Publicar, pertencentes aos casos de uso descritos no capítulo 3 que não foram implementados.

Compartilhar um projeto é permitir que outros professores usem o conteúdo de um projeto existente para uso próprio, sem alteração do projeto inicial. O objetivo do compartilhamento é tornar o sistema mais interativo e interessante aos usuários. O método de compartilhamento envolve a inclusão de atributos de permissão do projeto. Desta forma, apenas professores que são incluídos no projeto como planejadores ou executores possuem permissão para vê-lo, exceto quando este projeto é compartilhado. As permissões seriam públicas para todos os usuários registrados do sistema ou privadas para professores envolvidos com o projeto.

A publicação de um projeto envolve os botões “publicar” e “salvar”. O primeiro tornaria o projeto aberto a todos que possuem relação com o mesmo: professores planejadores, executores e todos os alunos que estão nas turmas as quais receberão o projeto. O botão salvar, por sua vez, salvaria no banco de dados as informações inseridas pelos professores planejadores, e somente estes poderiam alterar e ver o projeto. Na versão implementada neste trabalho, existe apenas o botão criar que libera a permissão a todos os envolvidos.

## 4.4 Atividades

Nesta seção serão detalhados os métodos básicos *CRUD* da classe Atividade.

A lista de atividades pode ser acessada pelo menu lateral por todo o sistema. Esta lista exibe os projetos separados em tipo plano ou execução para os professores e embaixo deles links para atividades que estão contidas nestes projetos, enquanto aos alunos é exibida apenas os projetos os quais suas turmas estão executando. Outra possibilidade de acesso de uma atividade única é através do projeto que as contém.

Na página da atividade é possível ver o botão para edição de uma atividade, a lista de materiais motivadores e um botão para criação de materiais. A página edição e a página de criação de atividades possuem apenas o campo de título necessário para definir o componente. O método de exclusão não foi implementado neste trabalho.

## 4.5 Materiais Motivadores

Nesta seção serão detalhados os métodos básicos *CRUD* da classe Material Motivador.

A lista de materiais motivadores pode ser acessada pelo menu lateral por todo o sistema. Na versão atual do SISLETRON optou-se por deixar aberta a lista de todos

os materiais motivadores contidos no banco de dados disponível para todos os usuários, pois não foi encontrada uma maneira elegante de exibir os materiais específicos de cada atividade para cada projeto em que o usuário está envolvido. Nota-se que trabalhos futuros devem optar por retirar esta lista completamente ou implementar uma versão mais elegante.

A página do material motivador exibe o título e o conteúdo como primeiro bloco seguido de um botão de editar para professores responsáveis. Em seguida, blocos de exercícios são dispostos de maneira a exibir somente os 140 primeiros caracteres para prevenir uma página longa e confusa. Cada um destes blocos de exercícios contém o link para a página correspondente. Ao final da lista, o botão para criação de exercícios pode ser encontrado.

**Material**

**História**

Os três porquinhos são animais que vivem felizes no campo. Eles possuem narizes grandes, coloração rosa e patas curtas. Na floresta ali perto mora seu predador natural. O nome dele é Lobo Mau e adora caçar porquinhos. Cada porquinho possui uma casa diferente. O primeiro porquinho tem uma casa de palha que aguenta três sopros do lobo. O segundo porquinho possui uma casa de madeira que aguenta cinco sopros. O terceiro e último porquinho possui uma casa de tijolo que aguenta nove sopros do terrível Lobo Mau (...).

**Editar Material**

**Exercícios**

Quais animais são descritos no texto e qual suas características?

Quantas vezes o Lobo Mau deverá soprar para derrubar as casas dos três porquinhos?

**Criar Exercício**

Figura 3: Captura de tela do sistema: tela exibição do material motivador para professores

A criação de um material motivador é acessada através do link contido na página de uma atividade. Este link coloca como parâmetro o id da atividade ao qual o material será vinculado e o insere na *URL*, como neste exemplo: `/material_motivadores/new?atividade_id=1`. Na página de criação do material, por sua vez, este parâmetro é enviado ao método *create* do controlador pelo campo escondido *hidden\_field\_tag(:atividade\_id, params[:atividade\_id])* da página HTML de criação. Este parâmetro é de fácil modificação pelo usuário e pode causar falhas de segurança, criando materiais dentro de atividades as quais o professor não tem permissão. Para corrigir este erro, uma verificação de permissões é feita no controlador

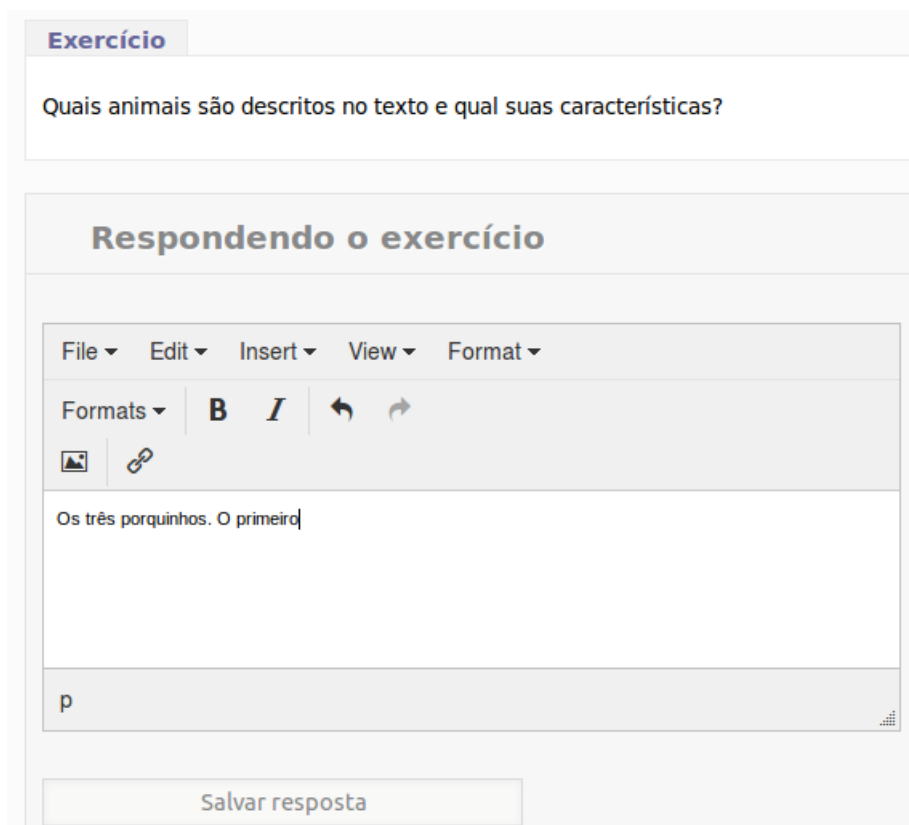
obtendo o id do projeto o qual o material está contido e comparando com as tabelas de execução e planejamento junto ao id do professor. Caso a permissão seja negada, o usuário retorna à página anterior e exibe o alerta adequado.

Nas páginas de criação e edição de materiais motivadores existem os campos de título e de conteúdo a serem preenchidos. O último é feito através do plugin de editor de texto *TinyMCE* apresentado na seção 2.2. Neste trabalho as ferramentas do plugin são restritas ao modelo básico do *TinyMCE* que contém seis tamanhos de cabeçalho e funções como negrito e itálico como principais métodos de edição de texto, além das funções de inserir links e imagens de fontes externas, sem a possibilidade de realização de upload.

O método de exclusão de materiais motivadores não foi implementado.

## 4.6 Exercícios

Nesta seção serão detalhados os métodos básicos *CRUD* e os métodos Corrigir e Realizar da classe Exercício



**Exercício**

Quais animais são descritos no texto e qual suas características?

**Respondendo o exercício**

File ▾ Edit ▾ Insert ▾ View ▾ Format ▾

Formats ▾ **B** *I* ↶ ↷

🖼️ 🔗

Os três porquinhos. O primeiro

p

Salvar resposta

Figura 4: Captura de tela do sistema: tela exibição do formulário de resposta para alunos

Os exercícios são criados e editados pelos professores responsáveis da mesma forma que os materiais motivadores, exceto pela ausência do atributo título. Após a criação, o usuário é redirecionado para a página do material motivador correspondente. Na página

de visualização do exercício, aos professores é exibido o botão para editar o exercício, enquanto aos alunos é exibido o botão de responder caso não exista resposta feita ou editar resposta caso contrário. O caso de uso Realizar Exercício visto no capítulo 3 é implementado de maneira a criar uma tabela de Respostas com o id do exercício e do aluno que o está realizando em conjunto com o conteúdo da resposta. O caso de uso Corrigir, por sua vez, foi implementado através da visualização das respostas feitas. Todas as respostas podem ser acessadas pelos professores responsáveis através de uma lista que contém um link com o nome, turma e escola do aluno que leva à visualização da resposta do exercício correspondente.

## 4.7 Classes Estáticas

Neste trabalho são chamados de classes estáticas aquelas que não possuem métodos de edição, criação ou exclusão. A visualização, por sua vez, foi implementada de forma parcial sendo somente exibida nas *views* de outras classes ou para testes. Estas classes incluem a Escola, Área de Conhecimento, Conteúdo e Turma. Elas existem no sistema para dar suporte lógico e prático às outras que foram consideradas mais importantes. No arquivo *seed* de população do banco de dados as classes são instanciadas com conteúdo relevante, entretanto essas informações não são editáveis ao longo do uso do sistema.

## 4.8 Casos de Uso não Implementados

Nesta seção serão descritos os casos de uso não implementados no sistema que podem ser considerados em trabalhos futuros.

### 4.8.1 Pesquisar Projeto

Esta funcionalidade tem como pré-requisito a implementação da publicação de projetos. Desta forma, projetos considerados incompletos ou privados pelos professores responsáveis não retornarão nos resultados das pesquisas. A pesquisa dos projetos pode possuir diversos campos de busca, como conteúdo, área de conhecimento, título e escola. Esta funcionalidade pode ser considerada de suma importância para o uso do sistema pois permite a interação de usuários de diferentes escolas sem a necessidade de possuir contato prévio.

### 4.8.2 Criar Dicionário

O caso de uso Criar Dicionário é necessário para que o professor possa trabalhar com as famílias de palavras, terminações de verbos, ordem alfabética, entre outros. Ou seja,

o professor cria um dicionário a partir das palavras encontradas no material motivador e aplica filtros nos dicionários criados, de acordo com o trabalho de Bueno (6).

Para criá-lo, deve-se implementar um novo modelo chamado Dicionário, construindo controladores e *views* adequados para armazenar as palavras na tabela do banco de dados e obter funcionalidades que permitam isto de forma fácil e intuitiva para o usuário.

### 4.8.3 Consultar Ajuda

Esta funcionalidade serve para que o professor e o aluno tenham um texto de apoio na utilização do sistema. Para a implementação precisa-se apenas da criação de uma página cujo link esteja contido no menu lateral ou no rodapé do sistema. Esta página conteria perguntas frequentes ou pequenos parágrafos explicativos.



## Conclusão

Neste trabalho foi possível observar a necessidade de criação de plataformas capazes de auxiliarem professores e alunos no processo de letramento infantil. Baseado no trabalho de Bueno (6) sobre letramento bilíngue para surdos e em conjunto com De Lara, o Sistema de Letramento Online “SISLETRON” foi projetado e desenvolvido com o objetivo de suprir esta necessidade.

Conforme descrito no capítulo 2, para a confecção da solução proposta neste trabalho utilizou-se o framework de desenvolvimento web *Ruby on Rails*. Este framework mostrou-se de grande valia na implementação do SISLETRON pois alivia a carga de trabalho associada a atividades comuns de desenvolvimento de um sistema web. Rails também auxilia na implementação de trabalhos futuros devido à baixa dificuldade de entendimento e de manutenção do código. Rails também possui gemas, bibliotecas que são integradas facilmente no sistema e que facilitam o trabalho de programadores.

No capítulo 3 foi detalhada a descrição conceitual do SISLETRON. Foi demonstrado na seção 3.1 o exemplo do projeto “Os Três Porquinhos” para melhor entendimento dos componentes necessários para a completa funcionalidade do sistema. Dentre eles, estão os usuários Professor e Aluno, o Projeto que contém diversas atividades, a Atividade que contém materiais base, Material Motivador que possui conteúdo que serve de base para o letramento de alunos, e Exercício que auxilia na fixação dos conteúdos presentes nos materiais. Na seção 3.2 detalhou-se cada componente mencionado anteriormente, exibindo seus modelos conceituais e os dividindo em classes e atributos. Na seção seguinte, os casos de uso definidos para a confecção do sistema foram apresentados. Observou-se que os casos de uso foram de suma importância para a realização deste trabalho pois guiou a implementação de forma a priorizar certas funcionalidades consideradas mais importantes.

O capítulo 4 foi responsável por descrever os detalhes da implementação utilizando os casos de uso descritos no capítulo 3. Devido ao tempo restrito na confecção deste trabalho o autor optou por implementar um conjunto restrito de casos de uso. Pelo mesmo motivo, algumas funcionalidades estão incompletas e necessitam serem revisadas em trabalhos futuros. Entretanto, foi possível finalizar o sistema de modo a permitir criação e edição na maior parte dos componentes, além da possibilidade de submissão de respostas pelos anos de exercícios criados pelos professores responsáveis.





## Referências

- 1 Instituto Paulo Montenegro. *Inaf 2011/2012 - Instituto Paulo Montenegro e Ação Educativa mostram evolução do alfabetismo funcional na última década*. Citado na página 7.
- 2 CHOMSKY, N. Language and problems of knowledge. *Teorema: Revista Internacional de Filosofia*, JSTOR, p. 5–33, 1997. Citado na página 7.
- 3 KLEIMAN, A. B. Os significados do letramento: uma perspectiva sobre a prática social da escrita. *Campinas, SP: Mercado de letras*, 1995. Citado na página 7.
- 4 SOARES, M. Novas práticas de leitura e escrita: letramento na cibercultura. *Educação e Sociedade*, SciELO Brasil, v. 23, n. 81, p. 143–160, 2002. Citado na página 7.
- 5 RAZET, C. De la lecture d’une histoire à la lecture d’une Écriture. *Synergies Brésil*, p. 59–74, 2012. Citado na página 7.
- 6 BUENO, J. *Pesquisa-ação na Construção de Insumos Conceituais para um Ambiente Computacional de Apoio ao Letramento Bilíngue de Crianças Surdas*. 2014. Citado 3 vezes nas páginas 7, 35 e 37.
- 7 FLÔRES MARIA LUCIA POZZATTI; TAROUÇO, L. M. R. R. E. B. Funcionalidades da ferramenta de autoria para apoiar a construção de objetos de aprendizagem. *Anais do XXII SBIE - XVII WIE*, 2011. Citado na página 8.
- 8 Web application framework. <[https://en.wikipedia.org/wiki/Web\\_application\\_framework](https://en.wikipedia.org/wiki/Web_application_framework)>. Acessado: 23/10/2015. Citado na página 9.
- 9 GUIDE, A. P. Agile web development with rails. 2006. Citado 2 vezes nas páginas 9 e 10.
- 10 FOWLER, M. *UML Essencial: Um breve guia para a linguagem-padrão de modelagem de objetos*. Tradução João Tortello. 3. ed. [S.l.]: Porto Alegre: Bookman, 2005. Citado na página 20.